

PHOTOREALISTIC RENDERING
UTILIZING CLOSE-RANGE PHOTOGRAMMETRY

by
SHAWN SLATER

A THESIS

Presented to the Department of Computer and Information Science
at the University of Oregon
in partial fulfillment of the requirements
for the degree of
Bachelor of Science

June 2016

THESIS APPROVAL PAGE

Student: Shawn Slater

Title: Achieving Computational Photorealism by Utilizing Close-Range Photogrammetry

This thesis has been accepted and approved in partial fulfillment of the requirements for the Bachelor of Science degree in the Department of Computer and Information Science by:

Hank Childs Advisor

Original approval signatures are on file with the Department of Computer and Information Science at the University of Oregon.

Degree awarded June 2016

THESIS ABSTRACT

Shawn Slater

Bachelor of Science

Computer and Information Science

June 2016

Title: Achieving Computational Photorealism by Utilizing Close-Range Photogrammetry

Approved _____

Hank Childs

The current process of asset creation in digital multimedia applications, such as movies and video games, requires tedious amounts of work. The average pipeline for creating one character model, for instance, takes an average of two weeks for the studio to implement, while still struggling to pass the uncanny valley for photorealistic rendering. Duplicating models from the real material world can speed up this process while also serving to preserve the intricate details that artists would otherwise overlook. Utilizing various applications, this paper details the use of photogrammetry as an option for the project pipeline of content creation in regards to photorealistic renderings for real-time applications. This paper will serve as a guide from planning to implementation of a human model into a real-time computer generated environment.

TABLE OF CONTENTS

Chapter		Page
I.	INTRODUCTION	5
II.	BACKGROUND	7
	3D Modeling Techniques	7
	Difficulties in Photorealistic Rendering	8
	Artist Errors	8
	What is Photogrammetry?	9
	Current Uses	9
	Types	9
	Photogrammetry Process	10
III.	METHODOLOGY	12
	Pipeline Overview	12
	Building the Set	12
	Model Preparation	13
	Camera Setup	14
	Shooting	14
	Photo Preparation	16
	Photogrammetry Software	18
	Mesh Retopology	20
	Texture Editing	23
IV.	ENVIRONMENT SETUP	25
	Real-time Rendering Engine	25
	Material Setups	25
	Lighting the Scene	25
V.	LIMITATIONS	26
VI.	CONCLUSION AND FUTURE WORK	27
VII.	MATERIALS / PRICING	28
VIII.	BIBLIOGRAPHY	29



Introduction

Creating realistic 3D art in a computer-simulated environment is difficult and time consuming. Artists do not usually look at models or references when creating their art. This not only shows in the topology of the structures created, but also in the textures painted to represent the details of the meshes. The project pipeline for game assets, for example, is usually a repetitive operation, where numerous assets are built to the best of the artists' abilities for the allotted time of construction. Assets commonly have to be recreated; therefore, the biggest battle for the artist is against time [1].

3D modeling software is a slow and tedious process, where the artist has to manually construct the object(s) with triangle, quadrilateral, or other (rarely used) polygonal constructs. To shorten the time of creation in the modeling of objects consisting of high poly count meshes, such as organic mammalian or reptilian models, the creations quite often are symmetrically equivalent. This commonly leads the artist to construct the objects along a bisection. This is then duplicated, mirrored upon itself, and then reattached as one piece to reduce construction time. As a result, the objects are neither realistic topologically or in detail [2].

After the construction of the bare meshes, numerous varied textures need to be mapped over the created topologies. Since the creations are usually referenced from the mind, they are painted and then applied to the surface models. Though it may be expedient, it is highly inaccurate, in regards to creating photorealistic pieces. There is, however, a faster route that is as accurate, if not more so, than many measuring and modeling methods [3].

Although using 3D modeling software has become the norm, using close-range photogrammetry not only expedites the modeling process, but can also be a means to create far more accurately-detailed textures for photorealistic renderings. In the following segments we will conduct a case study utilizing photogrammetry in the content creation pipeline as well as implement the asset into a real-time 3D environment for showcase. This paper aims to showcase the use of photogrammetry as an efficient option to be implemented, aside from the current manually created modeling infrastructures, in their asset creation pipelines for increased realism.



Background

3D Modeling Techniques

With today's advancements in technology, we have a large and diverse set of options to choose from for modeling objects. In the realm of manual content creation, polygonal meshes are commonly used. Polygonal modeling consists of object representations that are n-sided shapes defined by faces, vertices, and the edges between them, formed primarily of triangles or quads.

One option is to start with a simple primitive polygonal construct such as a cube or rectangle, where the artist may subdivide and extrude the object until the final shape is achieved [4]. One may also start the creation of an object by simply creating quad in succession, attaching each quad to another and modifying its position via the edges and vertices until the final shape is formed. These forms are manually created by vertex and line placements, utilizing Non-Uniform Rational Basis Spline (NURBS), which are used primarily for representing curves and surfaces mathematically. These mathematical representations of geometry, while having a few advantages over traditional modeling, in most cases, inevitably need to be converted to polygonal meshes for rendering, as the GPU cannot render them directly. This is often done via triangulation with the CPU, converting to polygonal data sets that the GPU can finally utilize for rendering. This excess in overhead, along with other deficiencies, makes polygonal modeling more ideal.

Digital sculpting is another option, which allows the artist to form their creations by making changes to an object in an organic way. This is often completed by use of a peripheral tablet/pen device, which is used to digitally carve and manipulate an initial clay-like slab, to form their desired model. This quite commonly utilizes voxel-based data sets for representation, though again, needs to be converted to polygonal meshes for use in the GPU, as current trends usually use the CPU to ray-trace the data, although GPGPU is a newer option for utilizing the GPU.

Outside of manual creation of 3D objects, there are also a range of options for content creation. Object data may be obtained from real-world objects via the use of images or lasers. The use of images to create 3D objects is referred to as “photogrammetry.” The differences between scanning 3D data, and taking images, both involve compromises. The creation of 3D objects through photogrammetry is considerably cheaper and less time consuming, compared to current 3D scanning techniques, however, scanning allows for far greater detail. Due to the limitations of the human eye, and even more, the graphical limitations of current visualization devices, photogrammetry appears to be the most promising source, in terms of logistics.

Content may also be automatically generated via “noise,” or other parameters, with the optional use of manually or automatically created assets, which is defined as procedural modeling. Entire terrains, clouds, vegetation, and populations can be created with near-infinite degrees of variations, on the fly.

Difficulties in Photorealistic Rendering

Aside from the physical and technological limitations associated with creating photorealistic renderings, there are issues of artistic skill. The Photorealism genre of art is defined by artists who study photographs and then strive to faithfully recreate it in another medium. While these artists achieve tremendous results, the created pieces are still lacking. Logistical issues for large scale content creation pipelines not only include the need to have an extremely large pool of these types of artists available, but also the financial constraints, and the amount of time it takes to create each piece would be and is currently absurdly insufficient.

Artist Errors

This situation creates the conditions where sub-par work is produced, in a trade-off to save on time and/or energy. This is not only a constraint for artistic resources; it is also a constraint for timeline milestones of production. With these compounding issues, financially and

logistically, we believe that photogrammetry may be more appropriate as a technique to be integrated into the content creation pipeline, for projects striving to achieve photo realistic quality. The artist no longer needs to strive for photorealism through their own means, as photogrammetry is by its very nature, photorealism.

What is Photogrammetry?

Photogrammetry is defined as “the art and science of extracting accurate geometric properties of objects from overlapping photographs” [5]. This process allows the automation of 3D content generation by triangulation of shared points from different photographs of the same object(s)/area [6]. Triangulation is the mathematical process of calculating lines and intersections, where this context is used to extract 3D coordinates. Once the process is complete, one is left with a finished or near-finished digital replica from the photographs.

Current Uses

Photogrammetry has numerous applications. This technology has been used for survey and mapping materials/applications. Museums catalog items, such as dinosaur bones, and various works of history, be it architecture, or smaller artistic pieces. Movies, such as *The Matrix*, used photogrammetry heavily for its production. Lately, video game companies have also been using this technology, such as, for the creation of the games, *The Vanishing of Ethan Carter* by *The Astronauts* [7], *Get Even* by *The Farm 51* [8], and *Star Wars Battlefront* by *EA* [9]. Aside from video games, Photogrammetry was heavily used in the creation of the hit movie, *The Matrix*, and more recently, used in the design of the *Digital Emily Project* [10].

Types

Primarily, two types are used: long-range and close-range. Long-range photogrammetry is

aerial, which is used to obtain geo-spatial data and building placements. The other type, close-range, is used to obtain fine details for reconstructing objects that are to be viewed up close.

Photogrammetry Process

The process of content creation utilizing photogrammetry is more of a game of patience. The amount of time it takes for sections to be completed largely depends on the number of photographs taken, the quality of the photographs, settings, and the power of the machine computing the information. However, once a system is set up and finely tuned, the bulk of the waiting time rests on the machine. This is not much of an issue though, as assets can be created in intervals.

The first part of the photogrammetry process will be obtaining photographs of the object(s) or area one wishes to recreate. Images are then used as input into a photogrammetry application. There are many out there; some free and some prohibitively expensive for hobby use. Some of the more expensive ones may be worthwhile for large scale projects that require greater detail and/or numerous assets to be created, as they cut down on creation time.

The software used will need to be able to have the images arranged by triangulation, which measures the distance of points on an image plane, compared to the camera plane. The shared points are reconstructed as a point cloud. Considering the large amount of points that may appear to be shared, the program will create an excessive number of needed points in the point cloud. Then a bundle adjustment algorithm is performed, which is commonly the Levenberg-Maquardt algorithm. The polygonal mesh is then created from the point cloud. Since photogrammetry in this context is being used to create photorealistic digital replicas, we will need a photo realistic texture to wrap around the mesh. Different texture map options are available, such as orthographic, spherical, etc., which needs to be applied to the mesh and then exported.

This will produce an asset that can be utilized as is, though other steps should be taken to

increase the quality of the asset. We can now recreate the topology of the asset in a third party 3D editing and creation application. The polygons should be restructured to smooth out imperfections, and also to have the poly count reduced to increase rendering performance. The texture may also have some noticeable blemishes that may need editing, depending on how the texture was recreated and/or applied to the mesh model.

While it may seem like a lot to do, as previously mentioned, it is primarily a waiting game. In this document, we will create a photorealistic bust from a human model. The pipeline will take us from the initial set building, to photo shooting the model, and through the editing process, to have a completed asset for use in a rendering environment. After we have created the asset, we will implement the asset in an integrated-development environment to be able to see our creation in action.



Methodology

Pipeline Overview

We will go over the photo-shooting process with the model after the set has been built. From there we will edit the images taken and input them into the photogrammetry software. After alignment, we will process the sparse and dense point clouds to be able to construct a highly detailed (and extremely high polygon count) mesh. Through edits, we will reconstruct the polygonal mesh and create the different maps (texture/diffuse, normal/bump, roughness/gloss/spec, and opacity/alpha). After we have refined our asset to our liking, we will implement the asset in a 3D real-time rendering engine.



Building the Set

While building a set is not necessary, it aids in the creation of the point clouds by excluding unwanted information introduced from the backgrounds of the model – hence it should not be part of the mesh. Not only will this reduce unwanted noise, it will also reduce the amount of time to do the calculations of the point cloud, sparse cloud, and final mesh, as there are fewer points to consider in triangulation. Lighting is also an important factor. Having a uniformly well-

lit environment helps to keep the colors corrected on the model that photos are being taken of. Without a uniform ambient light, shadows would also be introduced via natural ambient occlusion on the model [11]. One option is to plan for the model to be shot outside on an overcast day. This however cannot be well planned in advance, especially when taking into account the need for a static lighting environment over a long time period to produce enough images of acceptable quality. This, of course, is a natural option if the image shots will be taken outside for environmental models such as vegetation or rock formations.

For the model that I utilized (human), I decided that building a booth for shooting inside a building would be better suited for this project. This allowed me to control the lighting, and



allowed me enough time to take as many shots as I needed, without the worry of the environment changing. The booth I created consisted of four sheets, three white with slight transparency, and the fourth was a green screen [12] colored fabric. Allowing for enough space, I was able to direct the light from outside of the booth into it, through diffusion of the white sheet fabrics. I also left an opening in the front of the booth to allow light in to reflect off of the sheets/walls inside of the booth without having to worry about having any direct light hitting the model. For photographing small vegetation, such as leaves, a blue screen [13] can be used.

Model Preparation

There are many things to consider when preparing the model. If the model is a human, it's best for the person not to have any facial hair, as the small hairs could disrupt the overall creation of the object. For my model, I had asked him to come with a cleanly shaven face, and also to shave his head. Models with longer hair on their scalps may also wear a knit or bald cap to occlude the unwanted geometry. In regards to the skin for a human model, it should be relatively dry when taking the shots to reduce the amount of specular glare, as it would not allow for a

clean image. Such smoothness or oils/sweat may be recreated in the roughness/specular/glossy map at a later step. Aside from drying the skin, another option is to lightly dust a highly specular model with a powder (talc, makeup, etc) such as would be needed for a highly metallic material that is naturally glossy.

Camera Setup

Property	Value
Image	
Image ID	
Dimensions	6016 x 4000
Width	6016 pixels
Height	4000 pixels
Horizontal resolution	240 dpi
Vertical resolution	240 dpi
Bit depth	48
Compression	Uncompressed
Resolution unit	2
Color representation	sRGB
Compressed bits/pixel	
Camera	
Camera maker	NIKON CORPORATION
Camera model	NIKON D3200
F-stop	f/4
Exposure time	1/60 sec.
ISO speed	ISO-800
Exposure bias	0 step
Focal length	26 mm
Max aperture	4
Metering mode	Pattern
Subject distance	
Flash mode	Flash, auto, strobe return
Flash energy	
35mm focal length	39

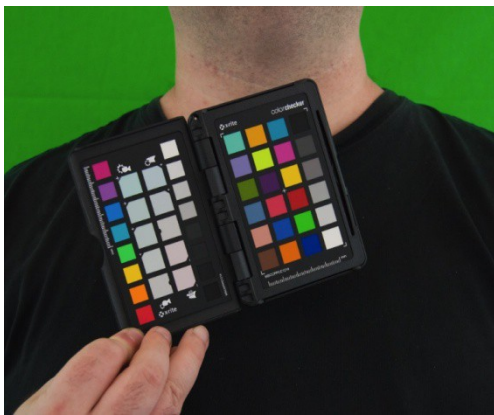
Camera set up is very important and should always strive for static settings. Considering that our set up consists of only one camera, this was not an issue. I also utilized a tripod for stability and to keep static distances from the object throughout the shots. The camera that I utilized was a Nikon D3200 DSLR 24.2 MP with a Nikon 18-55mm f/3.5-5.6G ED II AF-S DX Zoom-Nikkor Autofocus Lens [14].

It is important to have all of the shots be taken and recorded in a raw format, instead of other compressed lossy formats, such as JPEG, to not lose any information, as every pixel is important in the creation of the 3D asset. Keep in mind that these uncompressed files are rather large in size for today's standards. The following image displays the settings used for each of the 125 images taken (totaling 17 GB of data).

Shooting

Once you have your model in place for photographing, it is important to take reference shots with a calibration card, as I did with the Color Checker Passport created by X-right, which

was vital for calibration of the images [15][16]. Every time you photograph in a new environment (time or place) you will need to utilize the color checker for the best results. For the reference shots, I had the model hold the color checker passport in view for sampling and calibration. Shots with his eyes open were taken for future options of using the textures for the creation of the eyeball meshes, which could be created from sampling the image and/or used as reference for the other assets. For the purpose of this paper, no rigging, voicing, mesh animation or other art specific facets will be constructed outside of the basic creation of the model, as those are things that entire courses are taught on, and are outside the range of our scope.



For the shooting, it is best to take as many shots as you can. It is better to have too many than not enough, as the shoot takes a long time with one camera. And many of the shots may come out blurry, which are then unusable. We also want to get enough possible angles to construct the best three-dimensional image that we can from the triangulations.

Most importantly, we wanted a good overview of his face because, for this model, we will clothe him with a robe and hood, so the back top of the head, along with the ears, aren't really that important. More than 140 well-spaced shots were taken, including some close-up shots for finer details, though only 125 were kept due to quality issues. The model had to close his eyes and remain in the same position, aside from rotating on the stool, because when reconstructing the model, we can't have the model moving around, as that would not have an ideal outcome. Having the model move only on rotation was also acceptable, due to the use of the green screen, for quick masking. Otherwise, I would have had to have the model stay still as I moved around him. This would have required a larger, fully enclosed booth.

For static models, there is no worry about them moving, and you control all the angles and distances from the camera. This also allows you enough time to make sure the cameras are focused and you are getting a good quality shot. Also, for time constraints, aside from lighting environmental issues, you're allowed significantly more leeway, compared to working with a

model that may need to sit there for half an hour in the same spot, for good quality. Considering the limited amount of resources for this project, it was imperative that I had the best quality shots I could take, so it did take a while to go through the shoot.



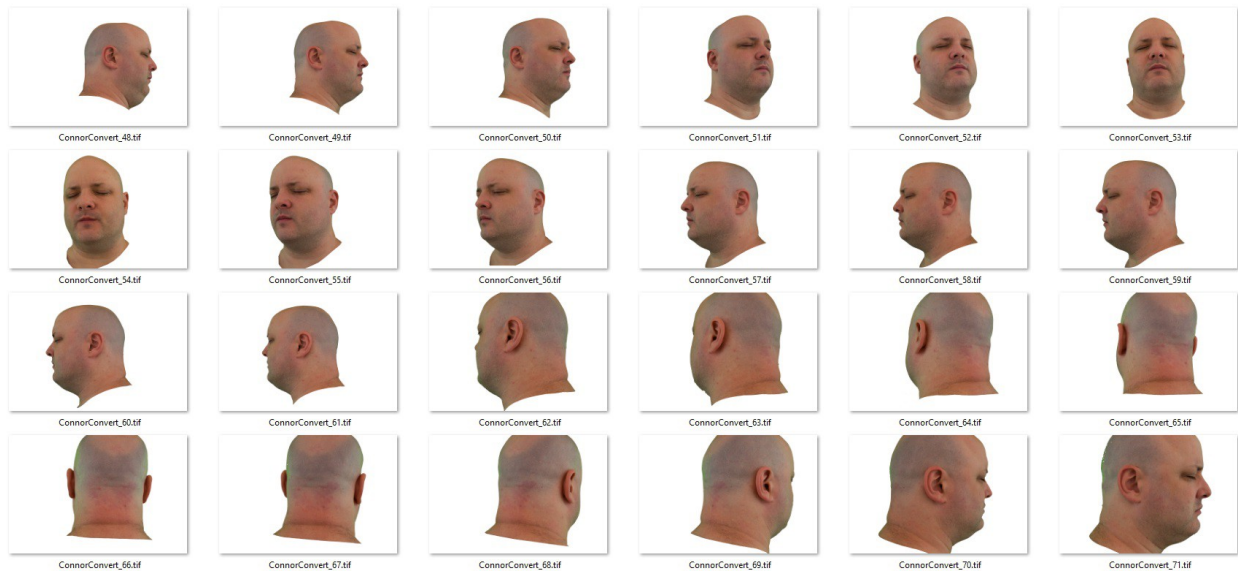
Photo Preparation

After all the shots have been taken, it is now time to prepare the photos to be processed for use in the photogrammetry software. The raw files in this shape can only be used by a small range of specific programs, so I will convert them to .DNG files via a free raw to dng file converter, provided by Adobe, although other programs exist, as well. The Color Checker Passport for calibration will come in handy here, as well as the past utilization of the green/blue screen for masking. The raw photos in their current state are placed in Adobe Light Room for image calibration, although there are many other options available, such as the software that comes with the Color Checker Passport. A color profile was created from the reference image with the passport in view, and the profile for colors and lighting was applied to all of the photos taken.

The next step utilized free software, called GIMP, for image editing, although the popular application, Adobe Photoshop, would work just as well. The goal now was to mask the photos and take out unwanted parts of the images, as some of the images had been ruined, partially due to global illumination of lighting bouncing off of the green screen onto the model. This is primarily on the sides of the model and not directly in the main focal-centered point of the

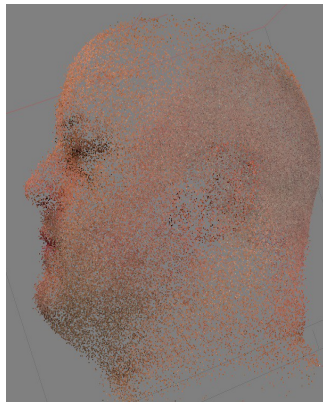
image. All of the green from the background, as well as the illuminated skin of the model, needed to be taken out, so that the green parts were not also aligned incorrectly with other parts of the image. That would have created a lot of static, and not have a good outcome. This was not too much of an issue, though, as there was still a significant amount of overlapping image area between shots, for triangulation.

After the good images were chosen, the background of green screen was very easy to select by color, given a high enough threshold, to delete out of the image. I chose to delete all of the green and make the background white. This made it much easier for the later process of having to auto select the background colors instead of having to manually go through each image and lasso select to mask [17] it which would be far too time consuming. All of the images were then saved as .TIF files, so not to have any loss in quality, and to be accepted into the photogrammetry application.



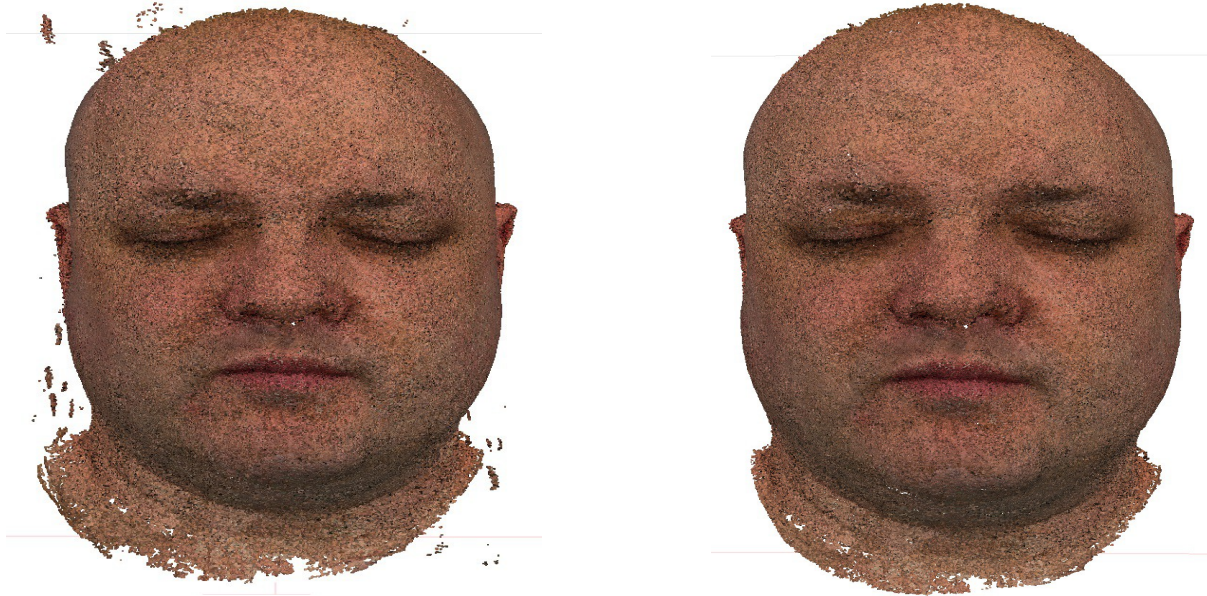
Photogrammetry Software

Now that all the files have been prepared, it is time to enter them into the photogrammetry software. The photogrammetry software I decided to utilize is a popular one, named Photoscan, by the company, Agisoft [18]. The images are first loaded into the software as one big chunk of many images. Masking the images is a quick process now that we had already prepped them.

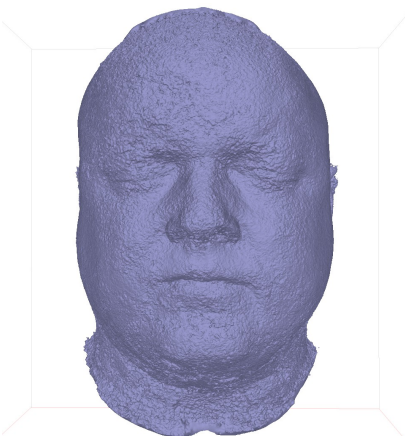


The photos are then aligned in preparation for the construction of the sparse cloud, as cameras. The sparse cloud is a low-density point cloud, which contains the low-quality vertex color points of the image to be created. After reviewing that the created sparse cloud has formed a decent image to our liking, that does not seem to need any other calibration of the cameras, then we can go ahead and create the density cloud. The density cloud is a high-density point cloud. Creation of this high-density point cloud will be the bulk of the time in the project pipeline, depending on the quality of your density cloud. For this project, the goal is to have the highest quality mesh to start with for editing, so the highest quality options for the density cloud were selected, which came out to be over 22 million points. This option on my computing desktop caused the program to run its calculations for about four hours for my 8 core CPU, 44 core GPU, 32 GB DDR3 RAM system, which is of decent standard for this time.

Now that this has been created, there may be a noticeable amount of noise introduced into the mesh, and it must be taken out. There will be points from the cloud that have been computed far from the original mesh, and they have to be tracked down, zooming out and panning around the mesh, highlighting with the collectible options, to delete. The bulk of the editing here will be from the points closer, finding other points that don't fit well within the mesh.



After you're satisfied with the overall density cloud, it is time to construct the mesh. The meshes created in this system are primarily triangles instead of quads, which are harder to work with, and require a lot more data. I chose a medium-sized mesh, which consisted of 1.5 million polygons/triangles. This is not a big issue, as we are going to change the structure of the mesh to be better to work with. From this point, you can either create a texture to apply to the mesh from the many options, or we can restructure the mesh. Applying the texture to such a large polygon mesh is pointless because it's going to need editing, and there's not a whole lot that we can edit with this mesh due to the current UV being exploded, instead of being properly unfolded.



From here, we're going to go to the menu and choose to smooth the mesh. This will make it a bit easier when redoing the topology (see the next section). One option here is also to not smooth out the mesh, though it might be more difficult to redo the topology. The



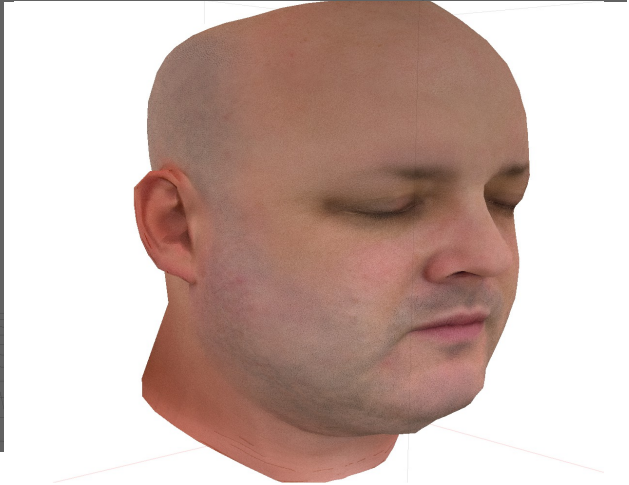
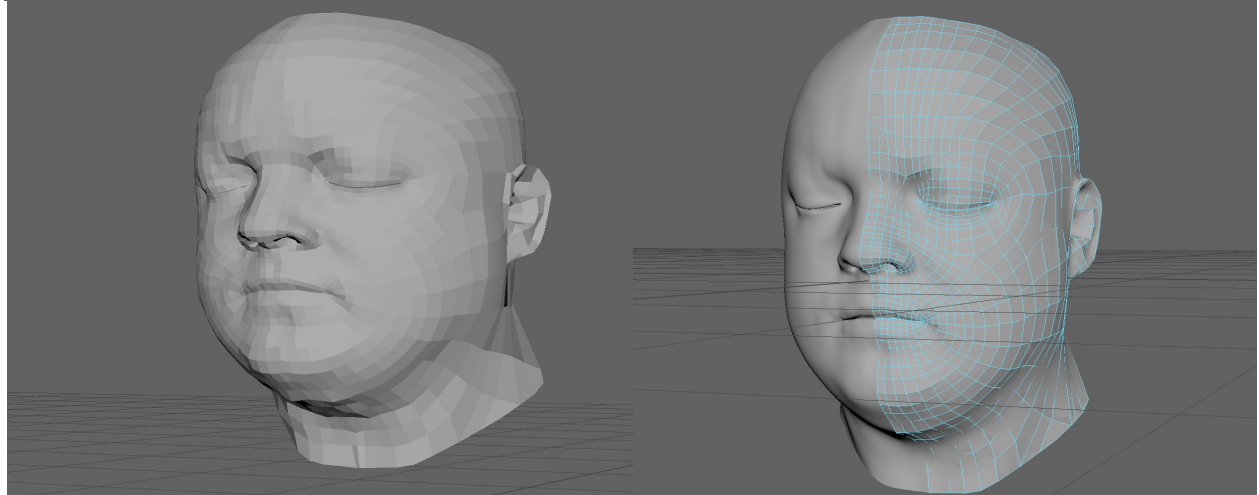
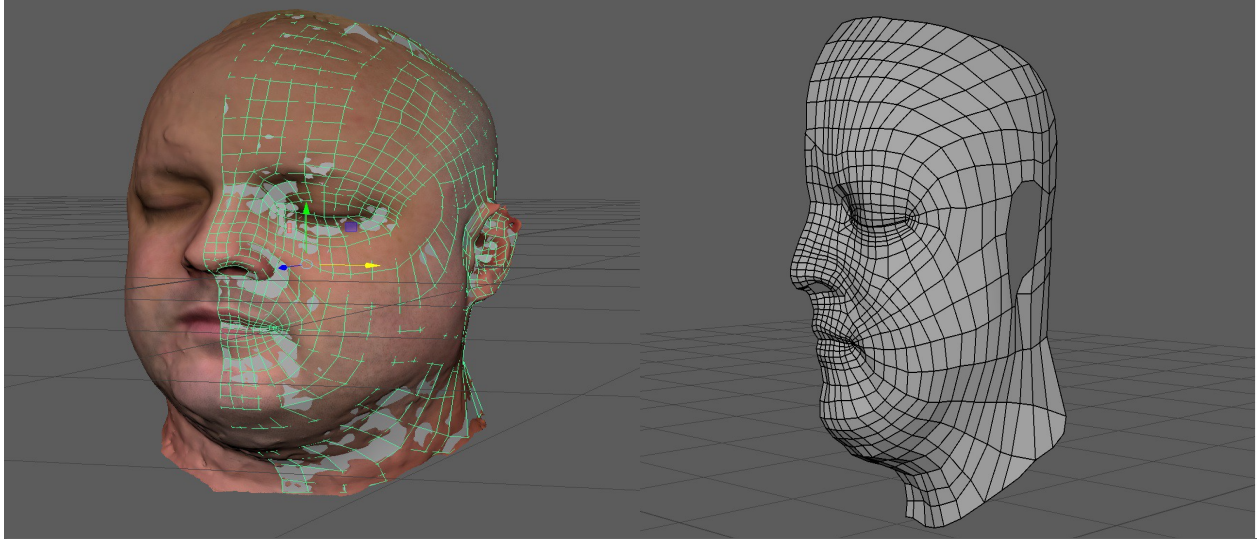
positive side of this is that you'll have more information to go off of when creating the normal maps for bumping, although we will

be better off in the long run to reconstruct, based off of the reference images. From here, we are going to export the mesh as a .OBJ file type. This file type is uncompressed and commonly used, so we may essentially put this in most, if not all, commercial/freeware 3D editing applications.

Mesh Retopology

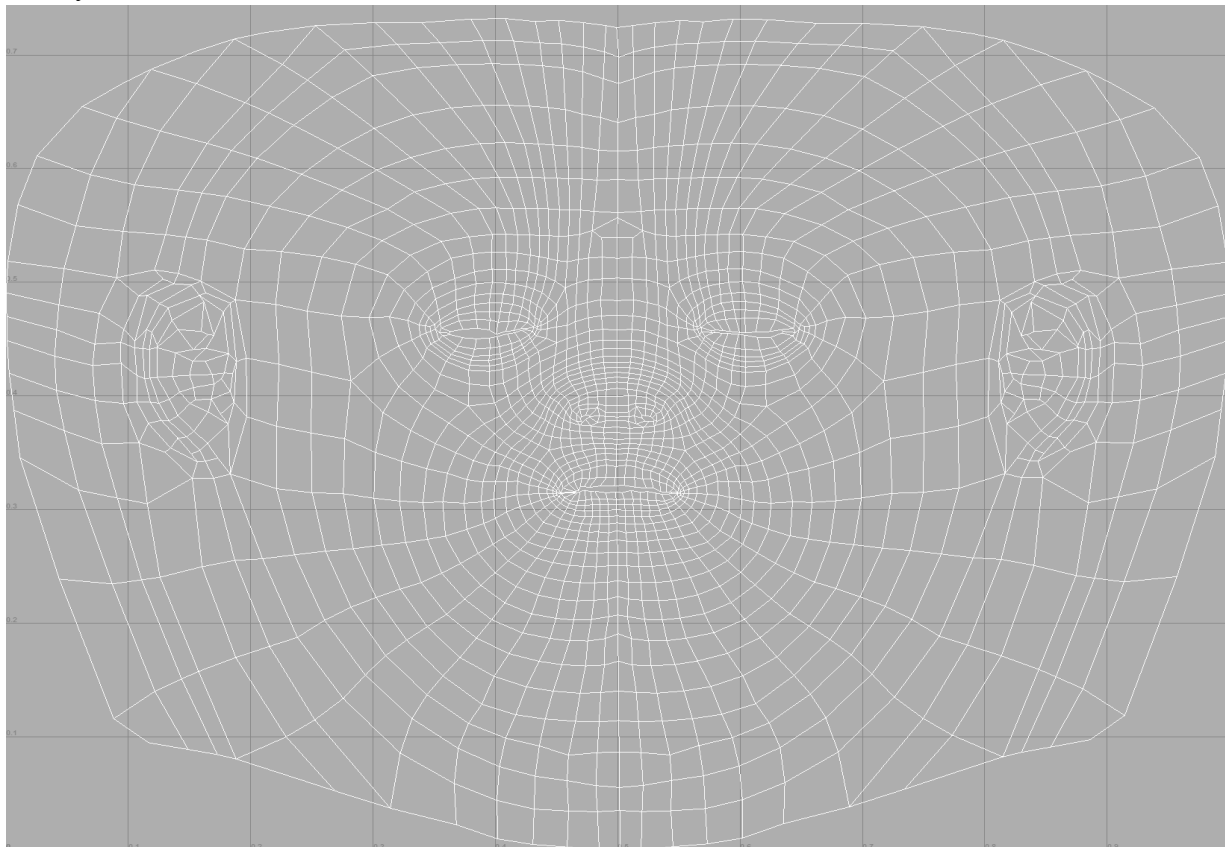
Retopologizing the mesh is the act of recreating the mesh's attributes (vertices, edges, faces, normals, etc) [19]. The reasons that the mesh needs a new topology is that there are too many triangles that need to be rendered at run-time and the UV layout needs to be reconstructed to be able to properly work with the applied maps (Texture, Normal, Roughness, and Opacity) that we will be using. While there are many options available to use to retopologize the mesh, I will utilize Maya by Autodesk. In Maya, I will utilize the quad-draw feature to recreate a new mesh over the old one, as seen in the image. This will also allow us to configure the textures better later, when they're created, allowing us more creative flexibility.

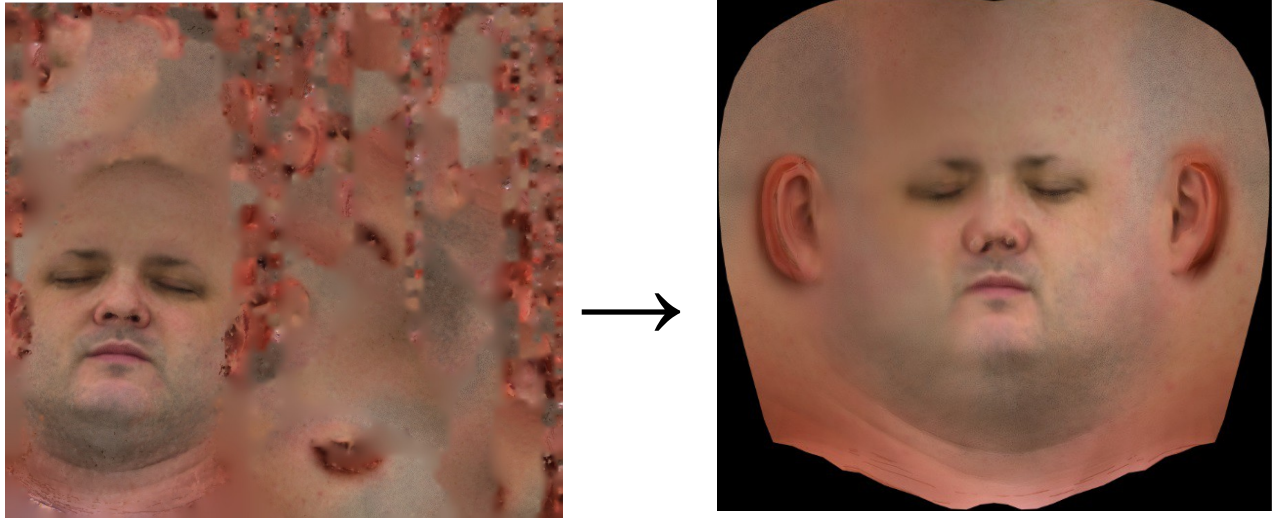




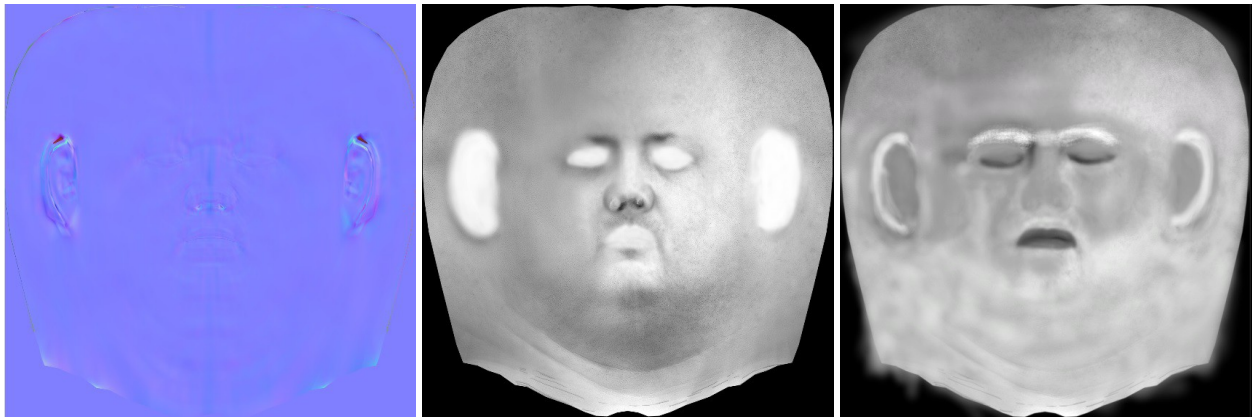
Now that the mesh has a new topology, the final mesh is roughly 3,000 triangles. From here, we will need to also recreate the UV layout. I am choosing a projection based on the front camera, where it is then unfolded. This is much easier to work with, as you can see. I also create another mesh by duplicating the old one, and smoothing it, which is about 14,000 triangles. This will be better used for later on, when we create the normal maps for finer details, to be transferred over to the lower poly mesh.

Now, from here, we could transfer the high poly mesh into a program, such as mud box or 3-D coat, or any other 3-D sculpting program able to create finer details on the high poly mesh, such as wrinkles, or raised surfaces on the model. This is quite common when creating human models. Once this is done, we can then create a normal map of the high poly model, and transfer that for the low poly model. Now that the topology has been recreated, and the UVs have been laid out accordingly, we can transfer the model back into the Photoscan application to finally create the texture, based on the newly imported mesh's UV layout. By deleting the old mesh, and importing the new one, we can re-center the asset, and choose to create a texture based on the UV layout.





As you can see now from the images shown, this texture is much easier to work with. I chose to use an 8K texture and other maps, however, the application we will use to render it in real-time only allows 4K textures, so it will automatically reduce them. I can keep them for more precise edits later, and also, some other engines allow for the 8K texture maps.



Texture Editing

Now that we have the textures, we can edit them for improving quality, so we will use the GIMP application, again. The first thing we are going to do is zoom in and fix any imperfections

that will show close up in the model. Due to the blended materials I will use in the engine, I am also going to create a duplicate texture map, and increase the contrast. We are going to need four types of maps. One will be a diffuse map, which is the main texture [20]. The second will be a normal map [21] for the lighting to display finer details that we want to have in the mesh (and have already created, but may edit it for wrinkles, etc.). The third will be a roughness map [22], which will be utilized to show even finer details around the normal and diffuse maps, to denote glossiness or roughness. The fourth and final one will be an opacity map [23]. In this example, this map is primarily used for subsurface scattering [24], which is needed to create realistic skin in a 3D environment.



Environment Setup

Real-time Rendering Engine

For the rendering environment, I decided to use the Unreal Engine [25], as this application has a lot of features, and pre-built skin material rendering examples, to make this process much quicker. Here's the image shown next to the example image they have, with the content examples of the application. This engine is one of the most advanced game engines for public use of our time.

Material Setups

The material setups for this application were already created in the example, although here is an image of the node network. This setup is built with the newer Physically Based Rendering technology for computer graphics [26]. Keep in mind that fine tuning would be needed to have the model look correct.

Lighting the Scene

There are many options for lighting the scene, as I show in one of these instances. I decided to use a scene already created and place in the character, as this documentation is on creating realistic images with photogrammetry, and not how to create environment scenes (Level design).



Limitations

Limitations I had for this project were primarily in lighting, as I had not created a very good lighting system. Brighter lights with better diffusion would have been more beneficial, although the lighting was acceptable for getting nice detail. As much as one-third of each image that I photographed did not encompass the entire view of the model, which meant there was a lot of wasted space. Not wasting space would have allowed for greater detail. The camera was comparable; however it could have also been better. More cameras stacked around the character model would have decreased the amount of time it took for photographs, and the possibility of ruined photos, due to the object making slight movements, due to rotation and natural movements of a person. I should've taken more photos than I had, to offer greater detail naturally from the creation of the mesh. When I created the booth, it was ad-hoc, mostly reaching out for objects I could attach the sheets to, and I would want to create a better studio that was more permanent for the long-term. Overall, though, for this project and the amount of time it took to create the assets, it worked out quite well.

Conclusion and Future Work

In conclusion, photogrammetry is an ideal way to create photorealistic renderings, especially when trying to duplicate existing models, which are very difficult to create by an artist, alone. While most high-quality characters take about two weeks to produce (Joel Lee – Former art instructor at DigiPen Institute of Technology), they can be made in a day or two utilizing photogrammetry. For future work, I would like to create full body scans of people, have them rigged, animated, and implemented in a full production for virtual reality systems. For large scale production, I would like to create a larger booth and utilize more cameras on a synchronized switch to save time and decrease the margin of error between shots due to the human models moving. In this project, I did not implement hair of any lengths. I would like to research this more as to what would be the best way to implement it. Perhaps using photogrammetry to obtain the color(s) of the hair(s) of the individual would be helpful in obtaining the correct values. Special material shaders would be required for this though just as was needed for the skin's subsurface scattering. I would have also liked to have had more knowledge of the Unreal Engine software. The program is quite powerful in my view compared to the Unity 3D software I have been using for the past few years. I plan to switch to using the Unreal Engine and continue using photogrammetry in my future projects.



Materials / Pricing

ITEM	COST
Nikon D3200 24.2 MP CMOS Digital SLR Camera with 18-55mm and 55-200mm Non-VR DX Zoom Lenses Bundle	\$483.38
X-Rite ColorChecker Passport	\$94.93
Twin Size Flat Sheets, T-180, white, 6-Pack	\$46.95
10 Feet X 10 Feet Cotton Chromakey Green Screen Muslin Backdrop Photo Photography Background	\$26.45
AmazonBasics 60-Inch Lightweight Tripod with Bag	\$21.49
SanDisk Ultra 64GB microSDXC UHS-I Card with Adapter	\$17.99
Nikon ML-L3 Wireless Remote Control	\$13.99
Transcend Information USB 3.0 Card Reader	\$5.99
TOTAL	\$711.17

SOFTWARE USED	COST
AutoDesk Maya 2016	Trial
AgiSoft PhotoScan 1.2.4	Trial
Adobe Lightroom 6.5.1	Trial
GIMP 2.8	Free
Epic Unreal Engine 4.12	Free

Bibliography

- [1] Pitzel, Steve. "The 3D Production Pipeline for Games: Getting It There." The 3D Production Pipeline for Games: Getting It There. Intel, 13 Sept. 2011. Web. 16 Apr. 2016. <<https://software.intel.com/en-us/articles/the-3d-production-pipeline-for-games-getting-it-there>>.
- [2] Watkins, Adam. Creating Games with Unity and Maya: How to Develop Fun and Marketable 3D Games. Burlington, MA: Focal, 2011. 270-273.
- [3] PhotoModeler. "PhotoModeler - Close-range Photogrammetry and Image-based Modeling." *PhotoModeler - Close-range Photogrammetry and Image-based Modeling*. [Http://www.photomodeler.com](http://www.photomodeler.com). Web. 04 June 2016. <http://www.faridesm.ir/jozavat/10_Photogrammetry_Common_Questions.pdf>. 10 Questions on Using a Camera for Measuring & Modeling: How photogrammetry software will change the way you work .
- [4] Bridgetharveysae. "Current Trends in 3D Modeling and Game Asset Pipeline Example." Bridgetharveysae. 02 July 2014. Web. 24 May 2016. <<https://bridgetharveysae.wordpress.com/2014/07/02/current-trends-in-3d-modelling-and-game-asset-pipeline-example/>>.
- [5] AEROMETREX. "Geospatial and photogrammetry: When the Mapping World and the Gaming World Collide." From High Above. Web. 24 May 2016. <<http://aerometrex.com.au/blog/?p=1052>>.
- [6] Foster, Shaun, and David Halbstein. Integrating 3D Modeling, Photogrammetry and Design. London: Springer London, 2014. 2.
- [7] Poznanski, Andrzej. "Visual Revolution of The Vanishing of Ethan Carter." The Astronauts. 25 Mar. 2014. Web. 24 May 2016. <<http://www.theastronauts.com/2014/03/visual-revolution-vanishing-ethan-carter/>>.
- [8] "Get Even." Get Even. Web. 24 May 2016. <<http://www.getevengame.com/>>.
- [9] "How We Used Photogrammetry to Capture Every Last Detail for Star Wars™ Battlefront™." EA Star Wars Games. 19 May 2015. Web. 24 May 2016.
- [10] Alexander, Oleg, Mike Rogers, William Lambeth, Jen-Yuan Chiang, Wan-Chun Ma, Chuan-Chang Wang, and Paul Debevec. "The Digital Emily Project: Achieving a Photorealistic Digital Actor." IEEE Comput. Grap. Appl. IEEE Computer Graphics and Applications 30.4 (2010): 20-31. Web. <<http://gl.ict.usc.edu/Research/DigitalEmily/DigitalEmily-IEEECGA-2010.pdf>>.
- [11] Foster, Shaun, and David Halbstein. Integrating 3D Modeling, Photogrammetry and Design. London: Springer London, 2014. 14-15.

[12] Hanke, Jeremy, and Michele Yamazaki. *GreenScreen Made Easy: Keying and Compositing for Indie Filmmakers*. Studio City, CA: Michael Wiese Productions, 2009. 7-8.

[13] Jackman, John. *Bluescreen Compositing: A Practical Guide for Video & Moviemaking*. Amsterdam: Focal, 2007. 14.

[14] Li, Zhilin, Jun Chen, and E. P. Baltsavias. *Advances in Photogrammetry, Remote Sensing and Spatial Information Sciences: 2008 ISPRS Congress Book*. Boca Raton (Fla.): CRC, 2008. 63-64.

[15] Ashe, Tom P. *Color Management & Quality Output: Working with Color from Camera to Display to Print*. Taylor & Francis, 2014. 101.

[16] Benjamin, Louis. *The Naked and the Lens: A Guide for Nude Photography*. 2nd ed. Focal, 2015. 210.

[17] Eismann, Katrin, Seán Duggan, and James Porto. *Photoshop Masking & Compositing*. Berkeley, CA: New Riders, 2013. Print.

[18] Ioannides, Marinos, Nadia Magnenat-Thalmann, Eleanor Fink, Roko Žarnić, Alex Yen, and Ewald Quak. *Digital Heritage: Progress in Cultural Heritage Preservation: Documentation, Preservation, and Protection: 5th International Conference, EuroMed 2014: Limassol, Cyprus, November 3-8, 2014 Proceedings*. 2014 ed. Springer, 2014. 403-404.

[19] Foster, Shaun, and David Halbstein. *Integrating 3D Modeling, Photogrammetry and Design*. London: Springer London, 2014. 3-4.

[20] Lengyel, Eric. *Mathematics for 3D Game Programming and Computer Graphics*. Boston, MA: Course Technology, Cengage Learning, 2012. 166.

[21] Lengyel, Eric. *Mathematics for 3D Game Programming and Computer Graphics*. Boston, MA: Course Technology, Cengage Learning, 2012. 178-186.

[22] Lengyel, Eric. *Mathematics for 3D Game Programming and Computer Graphics*. Boston, MA: Course Technology, Cengage Learning, 2012. 165.

[23] Möller, Tomas, Eric Haines, and Naty Hoffman. *Real-time Rendering*. 3rd ed. Wellesley, MA: A.K. Peters, 2008. 156, 181-183, 447, 482.

[24] Möller, Tomas, Eric Haines, and Naty Hoffman. *Real-time Rendering*. 3rd ed. Wellesley, MA: A.K. Peters, 2008. 225.

[25] "If You Love Something,." *What Is Unreal Engine 4*. Web. 24 May 2016. <<http://www.unrealengine.com/>>.

[26] Pharr, Matt, and Greg Humphreys. *Physically Based Rendering: From Theory to Implementation*. 2nd ed. Amsterdam: Elsevier/Morgan Kaufmann, 2010. Print.